

# 1 基于顺序存储结构的线性表实现

## 问题描述

为实现基于顺序存储结构的线性表及以此为基础的多表管理和文件读写功能，按如下形式设计数据的逻辑结构及基本运算。

### 线性表抽象数据类型

依据最小完备性和常用性相结合的原则，设计了线性表的数据对象和数据关系，并定义了线性表的初始化表、销毁表、清空表、判定空表、求表长和获得元素等12种基本运算，具体数据和运算功能定义如下。

**ADT** SqList {

**数据对象：**

**数据关系：**

**基本操作：**

SqList\_init(&L)

初始条件：线性表L不存在。

操作结果：构造一个空的线性表。

SqList\_destroy(&L)

初始条件：线性表L已存在。

操作结果：销毁线性表L。

SqList\_clear(&L)

初始条件：线性表L已存在。

操作结果：将L重制为 空表。

SqList\_empty(L)

初始条件：线性表L已存在。

操作结果：若L为空表则返回TRUE，否则返回FALSE。

SqList\_length(L)

初始条件：线性表L已存在。

操作结果：返回L中数据元素的个数。

SqList\_getElem(L, idx, &ele)

初始条件：线性表L已存在且非空， 。

操作结果：用ele返回L中第idx个数据元素的值。

SqList\_locateElem(L, val)

初始条件：线性表L已存在且非空。

操作结果：返回L中第一个值为val的数据元素的位序。若这样的元素不存在则返回0。

SqList\_priorElem(L, cur, &pre)

初始条件：线性表L已存在、非空且cur不是第一个数据元素。

操作结果：若cur是L的数据元素，则将它的前驱存入pre，否则返回FALSE。

SqList\_nextElem(L, cur, &next)

初始条件：线性表L已存在、非空且cur不是最后一个数据元素。

操作结果：若cur是L的数据元素，则将它的后继存入next，否则返回FALSE。

SqList\_insert(&L, key, val)

初始条件：线性表L已存在且非空， 。

操作结果：在L的第key个位置之前插入新的数据元素val。

SqList\_delete(&L, key, &val)

初始条件：线性表L已存在且非空， 。

操作结果：删除L的第key个数据元素，其值由val带回。

SqList\_traverse(L)

初始条件：线性表L已存在。

操作结果：依次访问并输出L的每个数据元素。

} **ADT** SqList

### 1.1.2 多线性表抽象数据类型

基于之前实现的线性表数据类型，设计了多线性表的……WTF?!

## 1.2 系统设计

下面有图：

图内容

图标题

下面有表：

表标题

表内容

下面有代码：

#include <stdio.h> // comments

#define DEBUG // 中文注释